## REMARKS

Claims 1-50 are pending in this application, all of which have been finally rejected. Claims 1-4 stand rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,237,135 (Timbol). Claims 5, 6, 8-10, and 13-16 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Timbol in view of U.S. Patent No. 6,424,979 (Livingston). Claims 7, 11, 12, and 17-49 have been rejected under section 103(a) as being unpatentable over Timbol in view of Livingston and in further view of U.S. Patent No. 6,654,029 (Chiu). Claim 50 has been rejected under section 103(a) as being unpatentable over Timbol, as modified in a manner proposed by the Examiner.

Applicants respectfully submit that there are several claim features that have been overlooked in the Office Action. In general, it appears that the Examiner regards the claims as being essentially the same as conventional JAVA, but has overlooked certain specific features recited in the claims that define over conventional systems. Applicants note that, on page 23 of the Office Action, the Examiner asserts that "Applicant [sic] has claimed basic JAVA component messaging, storing, and retrieval," and then goes on to list several features that are "well known."

With all due respect, applicants claims recite far more features than the Examiner's characterization suggests. Applicants would like to highlight several features that are not in any of the applied prior art, and request that the Examiner reconsider the rejection of the claims in view of the following features:

- separately retrievable customized code sequences (claim 1);

- selecting a code module from a plurality of code modules, where each of the code modules corresponds to a variable action (claim 9);

- retrieving or storing code, or a pointer to code, in a database (claims 5, 10, 17, 21, 27, 38, and 44);

- selecting a code module based on aspects of an environment in which the code module executes (claim 50 ) (similar features in claims 7, 12, 20, 32, and 33);

- a moniker string that is used as part of a query to retrieve code from a database (claims 12, 20, 32, and 33)

As to each of these features, the Examiner has either overlooked the feature, or has applied portions of the prior art that do not teach or suggest the feature. Applicants have demonstrated below specifically how these features differ from the applied prior art.

Claim 1 – separately retrievable customized code sequences

Claim 1 calls for a customization object that "invokes at least one of a plurality of customized code sequences based on said data or logic, each of the at least one customized code sequences being separately retrievable from a source separate from said customization object". The previous two office action apply this feature to Timbol's "java bean." The most recent office action argues that the "bean is separate from custom event set that is created and called by the bean object." However, a bean does not comprise "separately retrievable" code sequences. The bean described in the cited portion of Timbol is a unitary object, and no portion of Timbol cited by the Examiner – or any other that applicants have been able to locate – suggests that different code sequences within this unitary object can be separately retrieved. The claimed structure of separately retrievable code sequences is different from a single bean, since the single bean may have plural code sequences, but these sequences are not separately retrievable. Thus, applicants thus request reconsideration of the rejection of claim 1.

Claim 9 – selecting a custom code module from a plurality of code modules, where each code module corresponds to a variable action

Claim 9 recites "selecting a custom code module from a plurality of custom code modules," where each of the custom code modules comprises executable components, and where each of the executable components corresponds to a particular variable action. None of the applied prior art teaches this feature.

The crux of the Examiner's position on this point is found on page 19 of the Office Action, which states that Timbol's asserted teaching that "if the user wants the component to respond in some way to such an event, the user simply writes the code that responds within the body of the KeyPressed() method, for example." However, this quote has nothing to do with the claim feature of "selecting a custom code module," or with the feature of each code module corresponding to a particular variable action. The KeyPressed() method is not a

custom code module, but rather a method that is alterable by the user. While the KeyPressed() method arguably responds to a particular type of event, there is no indication that the KeyPressed() method is a module, or that it can be selected from a plurality of modules. On the contrary, the KeyPressed() method appears to be one of several functions that can be contained inside a single module. It does not appear from the applied prior art that there is any "plurality" of modules in respect to the KeyPressed() method. And, without a plurality of modules, there is nothing to "select." Thus, claim 9 is patentably distinct from the applied prior art, and applicants respectfully request reconsideration of the rejection of claim 9.

Claims 17, 27, 38, and 44 – retrieving or storing code, or a pointer to code, in a database

Claim 17 recites "retrieving, based on said database query, a code module from a database." According to the Examiner (see Office Action, p. 20), Timbol's teaching to the effect that "Application software can be any one of a variety of software application, such as ... database ...." Moreover, the Examiner asserts, without citation to Timbol, that "[c]ode modules are retrieved for loading." The fact that an application can be a database system, or that code modules can be retrieved for loading, is beside the point. Claim 17 does not call for an application that is a database; rather, claim 17 calls for the retrieval of code from a database. The rejection of claim 17 is based on the apparent assertion that Timbol teaches the retrieval of code from a database. While the cited portion of Timbol mentions the existence of a database, there is no applied teaching to the effect that such a database stores code, or that such code can be retrieved from the database. Nor has the Examiner explained any motivation to modify Timbol (or any other reference) to yield the feature of retrieving code modules from a database. Thus, the Examiner has not demonstrated that the features of claim 17 are obvious over the applied prior art, and applicants respectfully request reconsideration of claim 17.

Similarly, claims 27, 38, and 44 recite feature related to the storage of code (or a pointer thereto) in a database. The Examiner has not provided a detailed treatment of claims 27, 38, and 44; rather, claims 27 and 38 are addressed together with the rejection of claim 17, apparently without differentiating between claims 17, 27, and 38. Additionally, claim 44 has been addressed only in a single paragraph that covers claims 7, 11, 12, 18, 19, 20, 31, 32, and

44, without differentiating among these claims. Applicants cannot address how the prior art is being applied to the database features of these claims, since the Office Action does not explain which references, and which portions of those references, are being applied to the specific features recited in claims 27, 38, and 44.

Although it is not clear from the Office Action how the art is being applied to claims 27, 38, and 44, applicants note the following:

As to claims 27 and 38, these claims are treated together with are treated together with claim 17. Thus, for the same reasons discussed above in connection with claim 17, the applied prior art does not teach the database features of claims 27 and 38.

As to claim 44, the Examiner (at pages 10-11) asserts that Chiu discloses storing code in a database, but relies on a discussion of Chiu's asserted teaching of "data modeling." The Examiner has not explained how "data modeling" relates to the "database" feature. Applicants respectfully submit that the applied "data modeling" prior art has nothing to do with the features of claim 44 that relate to the retrieval of code, or a pointer to code, from a database. Thus, applicants respectfully request reconsideration of the rejection of claim 44.

For the foregoing reasons, applicants request that the rejections of claims 17, 27, 38, and 44 be reconsidered and withdrawn.

Claim 50 – selecting a code module based on aspects of an environment

Claim 50 calls for a code module to be selected based on factors comprising "aspects of an environment in which said code module executes." The Office Action, at page 17, asserts that Timbol teaches that "environmental features could be considered." However, in supporting this assertion, the Office Action relies on what is apparently the boilerplate language from the Timbol patent – i.e., Timbol's statement to the effect that "The present invention, however, is not limited to any particular application or any particular environment ... may be advantageously applied to a variety of platforms and environments." (See Office Action, p. 17, citing col. 5, ll. 62-66 of Timbol.) The fact that the quoted language mentions the word "environment" does not mean that Timbol teaches that a code module is selected based on "aspects of an environment in which said code module executes." In fact, the quoted passage merely says that Timbol can be applied in a variety of environments, but does not teach or suggest that Timbol selects a code module based on any aspects of such

environments. Thus, the Examiner has not demonstrated that this feature is taught in any of the applied prior art, and applicants respectfully request reconsideration of the rejection of claim 50.

Dependent claims 5, 7, 10, 12, 20, 21, 32, and 33

Several of the dependent claims recite additional features that are not taught in the prior art. A non-exhaustive list of such dependent claims includes claims 5, 7, 10, 12, 20, 21, 32, and 33, which recite features previously discussed above. For the same reasons discussed above, the applied prior art does not teach these features. In particular:

- Claims 5 and 10 calls for the retrieval of code from a database. As discussed above, none of the cited references describe the retrieval of code from a database.

- Claims 7, 12, 20, 32, and 33 call for the information to be derived from the environment, and for the derived information to be used as part of a database query. Except for claim 7, these claims all call for the information to be used to construct a moniker string that is used as part of the query. Timbol, Livingston, and Chiu do not teach these features.

- Claim 21 calls for the retrieval, from a database, of a pointer to code. As discussed above in connection with claim 44, none of the applied references teach the use of a database to store or retrieve a pointer to code.

In the section above that addresses the independent claims, applicants have discussed how these features differ from the applied prior art. Thus, in view of the arguments made, applicants respectfully request that the Examiner reconsider the rejection of these dependent claims in light of those features.

Conclusion

Claims 1, 5, 7, 9, 10, 12, 17, 20, 21, 27, 32, 33, 38, 44, and 50 have been shown to be patentable over the applied prior art, and the remaining claims are patentable at least by reason of their dependency. Thus, all claims have been shown to be patentable, and applicants respectfully request that the Examiner withdraw the remaining grounds for rejection and allow all pending claims.

Date: September 23, 2004

Peter M. Ullman
Registration No. 43,963

Woodcock Washburn LLP
One Liberty Place - 46th Floor
Philadelphia PA 19103
Telephone: (215) 568-3100
Facsimile: (215) 568-3439